

# Todo

Año 2 • Número 15 • 6,90 euros



# Programación

La Revista bimestral para entusiastas de la programación

www.studiopress.es



## Los pilares de Windows Vista

- ✓ Descubrimos la nueva plataforma WinFX
- ✓ Windows Workflow Foundation
- ✓ Windows Presentation Foundation
- ✓ Windows Communication Foundation
- ✓ XAML, un conjunto de XML especialmente diseñado para crear aplicaciones gráficas

**Trucos Visual C++:**  
Conectarse a Internet desde aplicaciones

**Novedades y cambios en MySQL 5 y PostgreSQL 8.1**

**Visual Basic para aplicaciones: Macros en Access**

**Publicación de bases de datos con OpenToro**

■ CD-ROM: ESPECIAL BASES DE DATOS (MYSQL, POSTGRESQL) Y ENTORNOS DE DESARROLLO PHP

### Zona Linux

- **Nemerle:** Conceptos básicos de la programación funcional
- **Mono:** Manejo de archivos XML con C#



### Zona Java

- **Struts:** Desarrollo de un ejemplo práctico
- **SUN Studio 11:** Programación para Linux/Solaris



ENTORNOS DE DESARROLLO PARA PHP, PHP DESIGNER, DEV-PHP, TULIP, GLADE, ETC.





## DIRECTOR

Eduardo Toribio  
etoribio@iberprensa.com

## REDACCIÓN

Yenifer Trabadela  
yenifer@iberprensa.com

## COLABORADORES

Antonio M. Zugaldia  
(azugaldia@iberprensa.com)  
David Santo Orcero  
(orcero@iberprensa.com)  
Manuel Domínguez  
(mdominguez@iberprensa.com)  
Fernando Escudero  
(fescudero@iberprensa.com)  
José Manuel Navarro  
(jnavarro@iberprensa.com)  
Marcos Prieto  
(mprieto@iberprensa.com)  
Guillermo "el Guille" Som  
(elguille@iberprensa.com)  
Santiago Márquez  
(smarquez@iberprensa.com)  
José Rivera  
(jrivera@iberprensa.com)  
Alejandro Serrano  
(aserrano@iberprensa.com)  
Jordi Massaguer  
(jmassaguer@iberprensa.com)  
Pedro Agulló  
(pagullo@iberprensa.com)  
Moisés Díaz  
(mdiaz@iberprensa.com)  
Fernando Marín  
(fmarin@iberprensa.com)  
Fabián Seoane  
(fseoane@iberprensa.com)

## DISEÑO PORTADA Y MAQUETACIÓN

Antonio G. Tomé

## DIRECTOR DE PRODUCCIÓN

Carlos Peropadre  
cperopadre@iberprensa.com

## ADMINISTRACIÓN

Marisa Cogorro

## SUSCRIPCIONES

Tel: 91 628 02 03  
suscripciones@iberprensa.com

## FILMACIÓN: Fotpreim Duvial

IMPRESIÓN: I. G. Printone  
DUPLICACIÓN CD-ROM: M.P.O.

## DISTRIBUCIÓN

S.G.E.L.  
Avda. Valdelaparra 29 (Pol. Ind.)  
28108 Alcobendas (Madrid)  
Tel.: 91 657 69 00

## EDITA: Studio Press

www.studiopress.es



## REDACCIÓN, PUBLICIDAD Y ADMINISTRACIÓN

C/ del Río Ter, Nave 13.  
Polígono "El Nogal"  
28110 Algete (Madrid)  
Tel.: 91 628 02 03\*  
Fax: 91 628 09 35

(Añada 34 si llama desde fuera de España.)

Todo Programación no tiene por qué estar de acuerdo con las opiniones expresadas por sus colaboradores en los artículos firmados. Los contenidos de Todo Programación son propiedad de Iberprensa y sus respectivos autores.

Iberprensa es una marca registrada de Studio Press.

DEPÓSITO LEGAL: M-13679-2004

Número 15 • Año 2

Copyright 1/04/06

PRINTED IN SPAIN

## EDITORIAL

### Windows Vista



Eduardo Toribio

La aparición de una nueva versión de un sistema operativo siempre supone una pequeña o gran revolución para los programadores dependiendo de unos cuantos factores. Principalmente los cambios que se introduzcan en el API o plataforma de desarrollo. Lo cierto es que, en principio, siempre tenemos una inercia negativa frente a los cambios, pero la realidad es que, guste o no, tendremos que ir pensando en migrar si queremos seguir desarrollando. Nosotros en este número hemos decidido abordar cómo va a afectar a los programadores Windows el nuevo API que nos ofrece la próxima versión del sistema de Microsoft.

Otros temas destacados que tratamos en este número son las últimas versiones de los dos gestores de bases de datos libres más importantes del momento, me refiero a MySQL y PostgreSQL. También analizamos Sun Studio 11, el novedoso entorno de desarrollo para Linux y Solaris y dedicamos talleres prácticos a OpenToro y Valgrind.

Finalmente, las correspondientes entregas de nuestros cursos dedicados a OpenGL, Netterrie, C#, PHP y VBA.

Espero que disfrutéis este número.

## SUSCRIPCIONES

Como oferta de lanzamiento existe la posibilidad de suscribirse durante 12 números a **Todo Programación** por solo 66,30 euros lo que significa un ahorro del 15% respecto al precio de portada. Además de regalo puedes elegir entre una de las dos guías que aparecen abajo. Más información en: [www.studiopress.es](http://www.studiopress.es)



## SERVICIO TÉCNICO

Todo Programación dispone de una dirección de correo electrónico y un número de fax para formular preguntas relativas al funcionamiento del CD-ROM de la revista.

e-mail: [todoprogramacion@iberprensa.com](mailto:todoprogramacion@iberprensa.com)

Fax: 91 628 09 35

## LECTORES

Comparte con nosotros tu opinión sobre la revista, envíanos tus comentarios, sugerencias, ideas o críticas.

## Studio Press

(Todo Programación)

C/ Del Río Ter, Nave 13  
Pol. "El Nogal"  
28110 Algete, Madrid

## DEPARTAMENTO DE PUBLICIDAD

Si le interesa conocer nuestras tarifas de publicidad no dude en ponerse en contacto con nuestro departamento comercial:

■ Tel. 91 628 02 03

■ e-mail: [publicidad@iberprensa.com](mailto:publicidad@iberprensa.com)





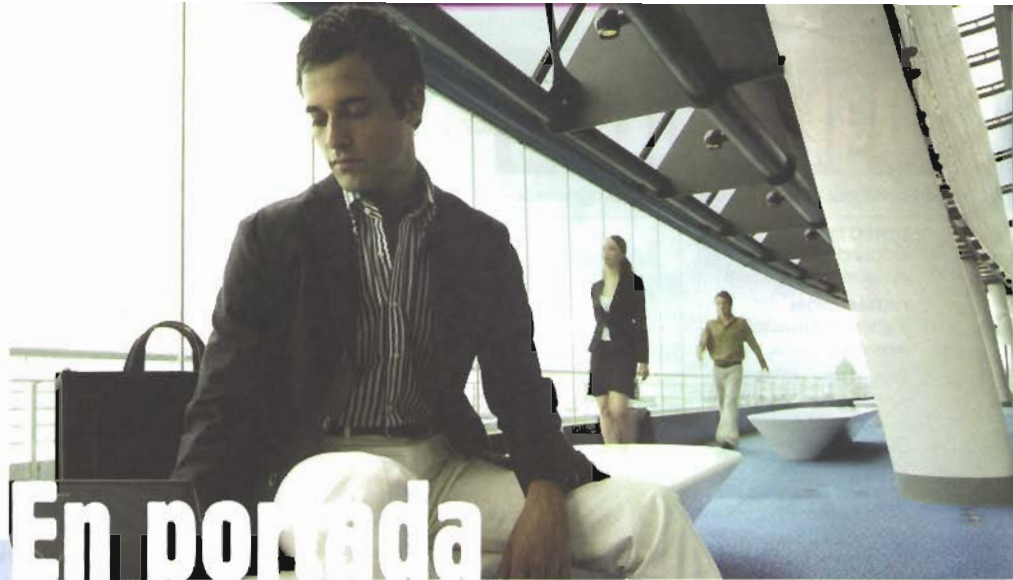
Número 15

### A quién vamos dirigidos

**Todo Programación (TP)** es una revista para programadores escrita por programadores y con un enfoque eminentemente práctico. Trataremos de ser útiles al programador, tanto al profesional como al estudiante. Si hay algo cierto en este sector es que nunca podemos parar, vivimos en un continuo proceso de reciclaje. Ahí es donde tratará de encajarse TP: información, actualidad, cursos y prácticas de los lenguajes más demandados y formación en sistemas.

### Ejemplos y código fuente

Cada CD-ROM de la revista incluye una carpeta denominada *fuentes* en la que se encuentra el material complementario para seguir cada uno de los cursos: por ejemplo, los listados completos, los ejemplos desarrollados en diversos lenguajes, compiladores, editores, utilidades y en general, cualquier herramienta que se mencione o cite en la respectiva sección. Todo con la finalidad de completar la formación y facilitar el seguimiento de cada artículo por parte del lector.



# En portada

## Los pilares de Windows Vista

**10** Windows Vista, el nuevo sistema operativo de Microsoft, supone una verdadera revolución en la plataforma. Aunque podemos seguir usando código nativo en C o C++ como hasta ahora, las verdaderas posibilidades vienen cuando utilizamos los lenguajes creados para .NET, especialmente C# y Visual Basic. Esta nueva API, denominada WinFX, consta de tres pilares fundamentales: WPF (Windows Presentation Foundation), el subsistema gráfico, dentro del cual se incluye XAML; WCF (Windows Communication Foundation), el nuevo sistema de comunicaciones que amplía las posibilidades de los servicios web, Remoting y otras tecnologías de conexión; y finalmente Windows Workflow Foundation (WWF), que permite crear gráficos de procesos que posteriormente se ejecutarán. Abordaremos todo esto y mucho más en nuestro reportaje de portada.



### ZONA LINUX

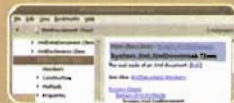
#### Actualidad: MySQL 5 y PostgreSQL 8.1 >>

**21** Analizamos las novedades y características que nos traen las nuevas versiones aparecidas de los dos sistemas gestores de bases de datos libres más populares: MySQL y PostgreSQL. Además, veremos el proceso de instalación y los primeros pasos a seguir con ambas herramientas.



#### Archivos XML con C# >>

**25** Cualquier lenguaje moderno requiere librerías para la manipulación de XML. .NET lo hace mediante el espacio de nombres *System.Xml*. En este taller aprenderemos a realizar las operaciones básicas con archivos XML: leer y obtener información, escribir datos y mezclar información de distintos ficheros.



## CONTENIDO DEL CD-ROM

### Herramientas y recursos para el programador

**64** En el CD de **Todo Programación** puedes encontrar en cada número las herramientas de programación más útiles y que conseguirán simplificar tu trabajo. Además, tendrás a tu disposición todas las aplicaciones y los ejemplos que se requieran para seguir correctamente los artículos y cursos que presentamos en la revista. En este número nos centraremos especialmente en los sistemas gestores de bases de datos y herramientas relacionadas.





## TALLER PRÁCTICO



### Publicación de bases de datos con OpenToro >>

**40** Es habitual tener una base de datos y querer realizar un interfaz para introducir y recuperar datos sin necesidad de SQL. OpenToro es una aplicación que nos ayudará a publicar bases de datos en la Web, generar listados, formularios de entrada, etcétera.



### El framework Struts: desarrollo de un ejemplo >>

**44** En esta segunda entrega de nuestra miniserie dedicada a Struts vamos a encarar el desarrollo de una pequeña aplicación de ejemplo en donde pongamos en práctica todas las ideas vistas, además de algunas otras nuevas que no comentamos en el pasado número.



### Valgrind >>

**50** En nuestro anterior taller introdujimos la depuración de errores con Valgrind. Ahora veremos cómo localizar algunos fallos de programación con esta herramienta a nivel práctico.



### OpenGL: Modelando el entorno >>

**55** En las entregas anteriores de nuestra serie aprendimos el funcionamiento básico de OpenGL, para posteriormente ejecutar nuestro primer programa con esta potente biblioteca. Ahora comprobaremos todas las posibilidades que ofrece a la hora de modelar.

## ZONA WINDOWS

### Trucos Visual C++: Conectarse a Internet desde aplicaciones >>

**28** Dedicamos nuestro espacio de trucos en VC++ a tratar diversas formas para lograr conectarnos a Internet desde nuestras aplicaciones, así podremos acceder a recursos en la Red tales como descargar archivos, ejecutar scripts o programas cgi.



### REPORTAJE: SUN Studio 11 para UNIX/Linux >>

**18** La programación en C/C++, lejos de quedar obsoleta, sigue estando muy en uso, sobre todo con la mayor popularidad de los sistemas UNIX/Linux. Esta circunstancia posiblemente anima a fabricantes como Sun a presentar nuevos entornos para desarrollo en sistemas UNIX, como es el caso de Sun Studio 11.



## Y ADEMÁS...

### Desarrollo Web: Entornos de desarrollo para PHP >>

**60** La aparición de los IDE (entornos de desarrollo integrados) supuso una revolución en



los métodos de trabajo para los programadores. Por supuesto, PHP no podía quedarse atrás y también dispone de varios IDEs que van a permitirnos optimizar nuestro tiempo y la gestión de los proyectos. Veamos a continuación algunos de los más utilizados.

### CUADERNOS DE PRINCIPIANTES

### Nemerle: Conceptos básicos >>

**32** En la entrega anterior de esta nueva serie nos introdujimos en la programación con el lenguaje Nemerle para la plataforma .NET. En esta ocasión aprenderemos a usar las estructuras de datos de las que nos provee, y conoceremos algunos conceptos básicos de la programación funcional.



### Visual Basic para Aplicaciones: programación en Access >>

**36** La gestión de bases de datos no es una tarea en absoluto sencilla, y sacar partido a Access implica no solo dominar este tema, sino también conocer la programación en VBA. Veamos una introducción a estos conceptos y algunos ejemplos.



### NOTICIAS

6. Sun dona software a los programas de red.es.
6. F5 lanza FirePass Controller.
7. IBM pone en marcha Latin America Grid.
7. Autodesk impulsa sistemas GIS libres.
8. Acuerdo Oracle-IBM.
8. Afina se convierte en Centro de Formación Oficial MySQL.
8. El Banco de España implanta la aplicación de cifrado de Secuware.
8. Soluciones HP OpenView.
9. Nuevo servidor Silicom Graphics Altix 4000.
9. Cámara Kodak EASYSHARE V570.
9. Servidores Sun Fire T1000 y T2000.





# Publicación de bases de datos con OpenToro

MANUEL DOMÍNGUEZ

mdominguez@iberprensa.com

**E**s habitual disponer de una base de datos y querer realizar un interfaz intuitivo para introducir y recuperar datos sin necesidad de crear sentencias SQL. Además, cada vez se desea más que este interfaz sea publicable en web. Sin embargo, las tareas de generación de listados o formularios son grandes consumidoras de tiempo.

OpenToro nos ayudará de forma excelente a publicar nuestras bases de datos en web, generar listados, formularios de entrada, etcétera, de forma sencilla y rápida.

En palabra de los propios autores, OpenToro es un gestor de contenidos web escrito en Java. Realmente es una aplicación que permite la publicación de una base de datos en una aplicación web J2EE. Esto hace posible olvidarse de escribir innumerables sentencias SQL o grandes scripts JSP, puesto que es OpenToro el encargado de abstraer al desarrollador de las particularidades de acceso a las bases de datos.

## REQUISITOS

OpenToro es, en sí misma, una aplicación J2EE. Por tanto, necesita un contenedor de aplicaciones J2EE donde ser desplegada. Tomcat es una buena opción, pero no la única. Podemos usar, por ejemplo, Application Server de Sun u otros. Además, necesita una base de datos sobre la cual residirán las tablas para las cuales OpenToro hará de recubrimiento, por lo que se requiere un SGBD instalado en el servidor. OpenToro soporta diversos SGBD: MySQL, Microsoft Access, Oracle y Microsoft SQL Server. A todos ellos accederá vía JDBC, como no podía ser de otra forma al estar desarrollado en Java.

## CONFIGURACIÓN DEL ENTORNO

El primer paso es descargar de la web de OpenToro (<http://opentoro.sourceforge.net>) todo lo necesario. En realidad no es mucho. Básicamente un fichero comprimido que contiene un WAR (incluido en nuestro CD-ROM) donde está almacenada toda la

aplicación y, no es obligatorio aunque recomendable, otro fichero comprimido donde se encuentra el tutorial que aunque breve, es de utilidad para los principiantes.

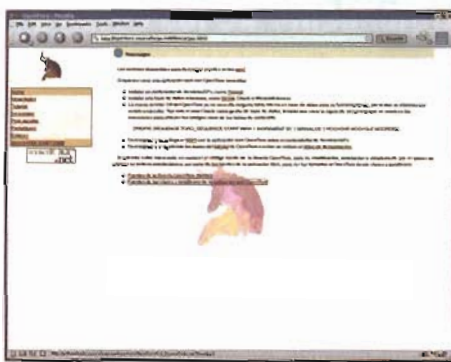


Figura 1. Sitio web de OpenToro.

El segundo paso es crear una base de datos en el SGBD que estemos utilizando para albergar las tablas a las que deseamos acceder en nuestra aplicación. Si ya tenemos dicha base de datos, entonces no es necesario nada más; En este taller se ha utilizado MySQL 4.0.15 para Windows y hemos llamado a esa base de datos *test* y después hemos creado un usuario con privilegios sobre ella, llamado *toro*, con su clave corres-

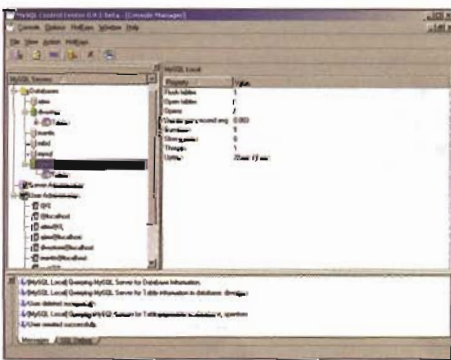


Figura 2. Creación de la base de datos a publicar.

pondiente. Estos datos serán necesarios después para configurar la aplicación.

Por último, necesitamos instalar un servidor de aplicaciones J2EE. Como no es ese el objetivo central de esta práctica, vamos a suponer que se tiene instalado y configurado correctamente Tomcat. Si no es así, el proceso es sencillo. Básicamente consiste en descargar de la web de Tomcat (<http://www.apache.org>) la última versión e instalarla haciendo doble clic en el archivo descargado. La instalación es muy elemental. Pues bien, si ya tenemos Tomcat instalado, solo tenemos que copiar el archivo *opentoro3\_spanish.war* en el directorio *webapps* del directorio de instalación de Tomcat. Y ya está (no está de más renombrar este fichero a *opentoro.war*). Reiniciamos el servicio de Tomcat y ya tendremos la aplicación OpenToro desplegada y la base de datos a la que acceder en MySQL. Ya no nos falta ningún componente. A partir de este momento podremos acceder a la aplicación tecleando en el navegador la siguiente URL: <http://localhost:8080/opentoro>.

## CONFIGURANDO LA CONEXIÓN DE OPENTORO

Una vez que tenemos la aplicación abierta en el navegador, debemos realizar unos ajustes para que pueda conectar a la base de datos que acabamos de crear, que será la base de información de OpenToro. Para ello debemos pulsar sobre la opción "Datos de conexión" que aparece en el recuadro central.

Y aparecerá entonces un formulario donde debemos especificar los datos de la conexión JDBC de OpenToro con su base de datos. Esta configuración es sencilla. En nuestro caso, especificamos como controlador JDBC la clase *com.mysql.jdbc.Driver*, que nos permite conectar con una instancia MySQL desde Java. Por otro lado, introduc-

## Las tareas de generación de listados o formularios consumen mucho tiempo



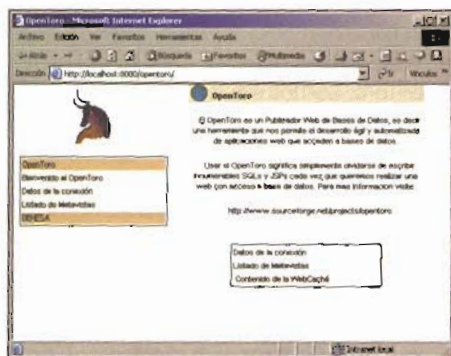


Figura 3. Página inicial de OpenToro.

mos la clave y el usuario de la base de datos que creamos unos pasos atrás y, por último, especificamos la URL de conexión a la base de datos; para nosotros, `jdbc:mysql://localhost/test`. Además, también es necesario indicar dónde se van a almacenar los ficheros XML que componen la metaprogramación de las vistas que crearemos (path relativo al directorio de despliegue de OpenToro), tras esto, pulsamos sobre "Establecer conexión". Aparecerá un mensaje en rojo indicando que la conexión se ha establecido correctamente y con esto habrá terminado la configuración de la aplicación para acceder a su base de datos.

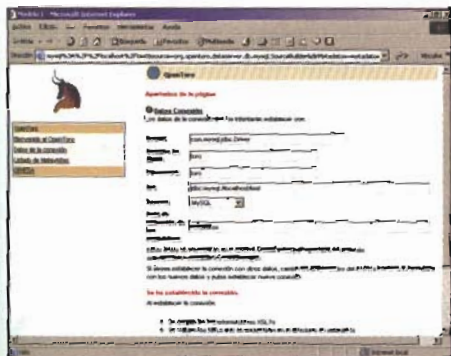


Figura 4. Conectar OpenToro a la base de datos.

## ■ FUNCIONAMIENTO GLOBAL

El funcionamiento de OpenToro es sencillo, pero a primera vista puede ser desesperante. Se basa en dos o tres conceptos clave: la metaprogramación de vistas, la llamada a métodos de los beans de OpenToro y la programación de JSP ligeros en nuestra aplicación web.

### Metaprogramación de vistas

Éste es el proceso más tedioso y desesperante de OpenToro. Consiste en describir en XML cada tabla o vista que queremos publicar, así como todos y cada uno de sus campos. Y como podemos realizar más de una metaprogramación para cada vista, pues en este caso habrá que incluir esta descripción XML también. En definitiva, OpenToro debe

conocer de antemano todos los detalles de cada tabla, campo y vista que deseemos publicar, para saber cómo debe actuar con respecto a ellas en cada caso y qué debe esperar. De esta forma, podremos abstraernos de ahí en adelante del acceso a esas tablas o columnas; y esta información se la suministramos en forma de fichero XML. Estos ficheros de metaprogramación se almacenarán en el directorio de 'metadatos' que hemos definido al establecer la conexión desde OpenToro.

Es un proceso realmente pesado, aunque sencillo, sobre todo para sistemas con un gran número de tablas o campos, pero si tenemos en cuenta que gracias a esa metaprogramación no vamos a tener que escribir ni una sola línea SQL ni accesos JDBC en nuestros JSP, vemos que a la postre resulta rentable.

### Llamadas a métodos de los beans de OpenToro

Una vez que hemos creado vistas metaprogramadas de las tablas o campos que deseamos consultar (tantas como se desee de cada tabla), OpenToro reconoce y puede trabajar con cada una de ellas e inmediatamente esas vistas están publicadas y pueden ser accedidas desde un JSP simplemente con la invocación de los métodos que nos ofrecen varios beans de OpenToro. Los métodos son de muy alto nivel y nos abstraen completamente de la arquitectura de la tabla. Por ejemplo, nos permiten obtener un formulario HTML para inserción de datos en una vista, listados HTML completos ordenados por cualquier campo y en cualquier orden del contenido de una vista metaprogramada (listados completos, listados paginados...); en definitiva, en lugar de hacer en un JSP una consulta JDBC a una tabla y posteriormente maquetar el resultado mediante más funciones JSP en pantalla, con una única línea de código hacemos una llamada a un bean de OpenToro y realizamos la misma operación. Esto es así gracias a la metaprogramación que hemos realizado de las vistas en OpenToro.

Otra de las ventajas adicionales que nos proporciona esta capa de abstracción es

que por ejemplo podemos tener una vista que nos devuelva los datos de una tabla ordenada por un campo y sin realizar cambios en el código, solo en la metaprogramación vía OpenToro, conseguir que los listados aparezcan ordenados por otro campo siempre. Es una filosofía muy potente.

### Programación de JSP ligeros

OpenToro no nos va a librar de tener que programar JSP en nuestra aplicación. Sin embargo, nos va a posibilitar crear vistas de una tabla incluso en SGBD que no permiten vistas, interponiendo una capa en medio (la metaprogramación) y por ello nos podemos olvidar de escribir ni una sola sentencia SQL, ni enmarronar el código fuente, ni maquetar el resultado de una `SELECT` en nuestra aplicación... etcétera. Por tanto, nuestros JSP se verán libre del 99 por ciento del código de acceso a datos, pudiéndose centrar exclusivamente en la lógica de negocio o en la capa de interfaz de usuario. De esta forma podremos tener JSP muy livianos, con poco código, fácil de mantener y fácil de depurar.

## ■ EJEMPLO ILUSTRATIVO

Una vez explicado el funcionamiento general, vamos a ver con un ejemplo paso a paso de cómo se metaprograman tablas y campos y cómo se accede a ellos desde una aplicación JSP.

### Creación de la tabla base

En la base de datos que llevamos utilizando durante todo el taller, vamos a crear una tabla típica llamada `PERSONAS` con un campo ID autonumérico que será la clave primaria, y los campos `NOMBRE`, `APELLIDO1` y `APELLIDO2`. Con esto tendremos lo que necesitamos para comenzar. El script SQL que genera esta tabla se puede ver en el Listado 1.

### Metaprogramación de la tabla PERSONAS

Para comenzar la metaprogramación de vistas o campos, es necesario dirigirse al apartado `DEHESA` de la ventana inicial de OpenToro. Digamos que OpenToro permite trabajar con las metavistas creadas y `DEHE-`

#### Listado 1

```
CREATE TABLE `personas` (
  `ID` int(11) NOT NULL auto_increment,
  `NOMBRE` varchar(50) NOT NULL default '',
  `APELLIDO1` varchar(50) NOT NULL default '',
  `APELLIDO2` varchar(50) NOT NULL default '',
  PRIMARY KEY (`ID`)
) TYPE=MyISAM;
```



SA es la parte de OpenToro que permite crearlas, validarlas, etcétera.

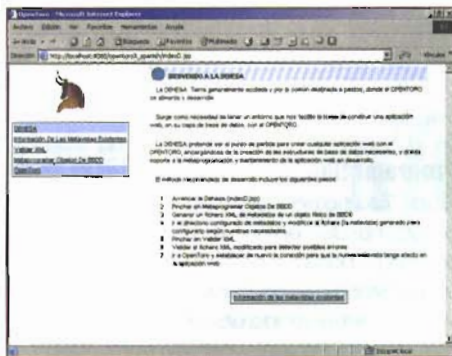


Figura 5. Vista de DEHESA.

Una vez estemos en DEHESA, hacemos clic sobre el enlace *Metaprogramar objetos de BBDD* que nos mostrará todos los objetos de la base de datos a la que hemos conectado OpenToro y nos permitirá metaprogramar cada uno de ellos. En nuestro caso, seleccionaremos la única opción posible que es la tabla *personas*, de nuestra base de datos.



Figura 6. Metaprogramar una vista.

Y se nos preguntará el código numérico que queremos dar a ese objeto metaprogramado. Esto es importante porque será el código que posteriormente tendremos que usar desde las páginas JSP para referirnos al objeto. Tecleamos el valor que deseamos y pulsamos sobre "Generar fichero". Esto creará una metaprogramación por defecto para la tabla *personas* que, en el disco, es un fichero XML que estará en el directorio *meta-datos* que configuramos anteriormente. Hasta aquí llega DEHESA (ya es mucho). Ahora cualquier personalización de esa metaprogramación que deseemos realizar ha de ser manualmente en el fichero generado. Aún así, en la opción *Validar XML* de DEHESA, podremos comprobar que no hemos "metido la pata" en esta operación. Y, generalmente, las personalizaciones se reducen a copiar y pegar y cambiar dos o tres parámetros. En cualquier caso, los cambios que deseamos hacer deben estar sujetos al esquema XML utilizado para validar las metaprogramaciones, que se puede ver en *WEB-INF/schemafile* y que define sin lugar a dudas todos los pormenores. Como ayuda, en la **Tabla 1** se muestran los nodos XML

más usuales del fichero de metaprogramación, aunque habrá que dirigirse siempre al esquema XML para conocer la obligatoriedad y cardinalidad de ellos, entre otras cosas.

Cuando hayamos terminado de modificar la metaprogramación (si es necesario y lo deseamos), debemos ir a DEHESA y validar el fichero XML que hemos modificado. Si todo está correcto, lo cual nos aparecerá claramente en el interfaz de usuario, volvemos a OpenToro y establecemos de nuevo la conexión para que la aplicación pueda recargar los datos de las metavistas que hemos creado y publicarlas. Y con esto termina el proceso de metaprogramación de objetos, que es la parte inicial.

## CREACIÓN DE UNA APLICACIÓN WEB

Para ver la verdadera utilidad de OpenToro es necesario crear una aplicación web que acceda a las metavistas publicadas por éste y en la que podamos apreciar cómo con un par de líneas se acceden a los datos para inserción, listado, modificación, etcétera. En las siguientes líneas continuaremos el ejemplo con unas series de páginas JSP demostrativas que accedan. Antes de nada, cargaremos algunos datos (los que queramos) en la tabla *personas* para poder trabajar.

### Listado de personas

En el **Listado 2** de la página siguiente podremos observar el script JSP necesario para hacer una consulta a las metavistas con idea de obtener un listado. Para ello se hace uso del bean *listWebBean*, que se encarga de obtener un listado de las metavistas y puede ser parametrizado de distintas formas. Este

bean tiene diversos métodos, que no están demasiado bien detallados en el manual de OpenToro. Son los siguientes:

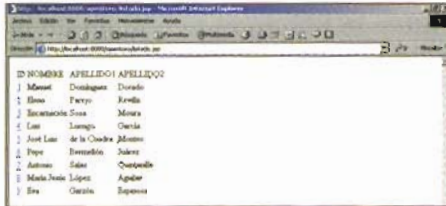
- **addFilter(filtro, dominio, valor):** que permite que el listado aparezca filtrado. *Filtro* es el nombre de la columna que queremos filtrar de la metavista, *dominio* es un operador tipo mayor, menor, igual... y *valor* es el valor (valga la redundancia) con el que queremos comparar.
- **addTextualFilter(clausulaWhere):** que permite filtrar pero mediante el uso de *LIKE*. Por ejemplo, habría que pasarle algo como *'WHERE NOMBRE LIKE %PEP%'*.
- **addParameter(parametro, valor):** permite pasar parámetros a las plantillas XSLT encargadas de generar el código HTML final.
- **getList(Plantilla):** devuelve el HTML que estamos creando y que será el que mostraremos, presumiblemente, a nuestros clientes. El parámetro debe ser un valor numérico cuyo valor no se especifica en el manual de OpenToro, pero que se puede ver en el código fuente de la aplicación en el fichero *listWebBean.java*.
- Y por último, **getPagesList()** nos devuelve el índice en HTML con los enlaces a las páginas si estamos sacando un listado paginado (esto se define con parámetros). Es cómodo para permitir a los clientes acceder a listados largos de forma cómoda.

De esta forma, para ejecutar el script de listado que hemos creado habría que teclear en el navegador *http://localhost/opentoro/*

## Tabla 1

NODO XML	ÁMBITO	DESCRIPCIÓN
<b>TableName</b>	Tabla	Nombre de la tabla.
<b>TableDesc</b>	Tabla	Descripción de la tabla.
<b>TableIndexField</b>	Tabla	Clave primaria de la tabla.
<b>FieldName</b>	Campos de la tabla	Nombre del campo.
<b>FieldAlias</b>	Campos de la tabla	Alias que se va a asignar en la sql al campo.
<b>FieldDescription</b>	Campos de la tabla	Nombre del campo en los listados y formularios.
<b>FieldType</b>	Campos de la tabla	Tipo del campo, según los valores definidos por OpenToro.
<b>FieldRequired</b>	Campos de la tabla	Campo requerido o no cuando se muestre en un formulario.
<b>FieldLength</b>	Campos de la tabla	Indica la longitud que permite el campo en la BBDD.
<b>MvDescription:</b>	Metavistas	Descripción para la metavista.
<b>MvTitle:</b>	Metavistas	Título de la metavista que va a aparecer en los listados y formularios.
<b>MvOrderField:</b>	Metavistas	Orden del campo en los listados.
<b>MvComboField:</b>	Metavistas	Campo que se mostrará en los campos automáticos.
<b>MvGroupByExpr:</b>	Metavistas	Permite incluir una cláusula <i>group by</i> al SQL.
<b>MetaViewList:</b>	Metavistas	Campos que queremos mostrar en un listado.
<b>MetaViewRecord:</b>	Metavistas	Campos que queremos mostrar al ver un registro.
<b>MetaViewForm:</b>	Metavistas	Campos que queremos mostrar en un formulario.





ID	NOMBRE	APELLIDO
1	Manuel	Domínguez
2	Blanca	Pérez
3	Isabel	Sanja
4	Lucas	Sanja
5	Jose Luis	de la Cruz
6	Pepi	Domínguez
7	Antonio	Sanja
8	María José	López
9	Eva	García

Figura 7. Listado de la metavista personas.

listado.jsp?table=1000. Suponiendo que a la metavista de la tabla personas que creamos le asignamos el código 1000, claro. Nos aparecerá el listado como se muestra en la Figura 7.

### Vista de un registro

En el Listado 3 (que se encuentra en el CD-ROM de la revista) podemos observar el script JSP necesario para hacer una consulta a las metavistas con idea de obtener el contenido de un registro concreto de una metavista. Para ello se hace uso del bean *recordWebBean*, que se encarga de obtener un listado de las metavistas y puede ser parametrizado de distintas formas. Este bean tiene diversos métodos, que de nuevo no están demasiado bien detallados en el manual de OpenToro. Son los siguientes:

■ **addFilter(filtro, dominio, valor):** que permite que el contenido del registro aparezca filtrado. *Filtro* es el nombre de la columna que queremos filtrar de la metavista, *dominio* es un operador tipo mayor, menor, igual... y *valor* es el valor (valga la redundancia) con el que queremos comparar.

■ **addParameter(parametro, valor):** permite pasar parámetros a las plantillas XSLT encargadas de generar el código HTML final.

■ **getContent(Plantilla):** devuelve el HTML que estamos creando y que será el que mostraremos, presumiblemente, a nuestros clientes. El parámetro debe ser un valor numérico cuyo valor no se especifica en el manual de OpenToro pero que se puede ver en el código fuente de la aplicación en el fichero *recordWebBean.java*.

■ **getEmptyForm(Plantilla):** nos devuelve el HTML correspondiente al formulario necesario para introducir datos del tipo que queremos consultar.

■ Y por último, **getFilledForm(Plantilla)** devuelve el mismo formulario pero completado con los datos de un registro concreto (caso de listado propuesto).

### Listado 2

```
<%@ include file="/connection/connection.jsp" %>
<jsp:useBean id="listWebBean" scope="session"
class="org.opentoro.modelo1.beans.ListWebBean" />
<jsp:setProperty name="listWebBean" property="request" value="%{request}" />
<jsp:setProperty name="listWebBean" property="connFact"
value="%{(ConnFactoryWrapper)request.getSession().getServletContext().get
Attribute(\"connFactoryWrapper\")}" />
<jsp:setProperty name="listWebBean" property="page" value="1"/>
<jsp:setProperty name="listWebBean" property="range" value="10"/>
<jsp:setProperty name="listWebBean" property="paginacion" value="true"/>
<jsp:setProperty name="listWebBean" property="*" />
<html>
<head>
</head>
<body>
<table>
<tr>
<td>
<%
String sContent = "";
sContent += "<p>";
sContent += listWebBean.getList( listWebBean.LIST );
sContent += "</p>";
out.print(sContent);
%>
</td>
</tr>
</table>
</BODY>
</HTML>
```



ID	NOMBRE	APELLIDO
4	Luc	Sanja

Figura 8. Vista de un registro.

De esta forma, para ejecutar el script de listado que hemos creado habría que teclear en el navegador `http://localhost/opentoro/unregistro.jsp?table=1000&element=4`. Suponiendo que a la metavista de la tabla personas que creamos le asignamos el código 1000 y tenemos al menos cuadro registro, claro. Nos aparecerá el listado del registro como vemos en la Figura 8.

### Formulario de inserción

Del mismo modo que el punto anterior, se utiliza el bean *recordWebBean* para obtener formularios de entrada de los datos. Solo con aplicarles una hoja de estilos adecuada, aparecerá con el aspecto que nosotros deseemos. En el Listado 4 (en el CD-ROM de la revista), se puede observar el código necesario para poder crear un formulario (una sola línea JSP! De este modo resulta muy sencillo construir aplicaciones

Para ejecutar el script de listado que hemos creado habría que teclear en el navegador `http://localhost/opentoro/entrada.jsp?table=1000&element=4`. Suponiendo que a la metavista de la tabla personas que creamos le asignamos el código 1000. Nos aparecerá el formulario que observamos en la figura.

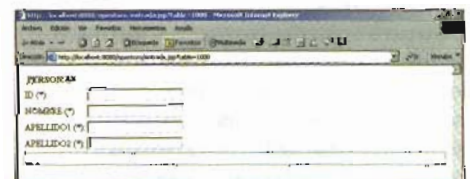


Figura 9. Creación de un formulario web.

### CONCLUSIONES

OpenToro es una herramienta robusta y que permite un desarrollo rápido con poco aprendizaje. Las sentencias SQL necesarias en sistemas complejos son difíciles de modificar y depurar, al igual que los JSP donde se entremezcla el código HTML, la lógica de la aplicación y el acceso a los datos. OpenToro nos elimina la mayor parte de estos problemas de un plumazo y permite que el desarrollador se centre en lo realmente importante, en desarrollar la aplicación. Es software libre, está bien diseñado, es eficiente y es una aplicación J2EE (con más proyección empresarial), por lo que no hay ninguna razón para no usarlo y para no beneficiarse de sus enormes ventajas.



1ª entrega, DVD-ROM  
y presentación de la obra  
por solo **5,99** euros



# USUARIO LINUX



► **Usuario Linux** es una obra compuesta por doce entregas de periodicidad semanal, cuyo objetivo es formar al lector en el manejo del sistema Linux y sus principales aplicaciones

► Aprende a trabajar de manera práctica, con tutoriales guiados y explicaciones sencillas, con el sistema operativo de mayor futuro, la plataforma que ya utilizan las empresas

► Linux es el sistema más moderno y eficiente para PC, con él olvidate de cuelgues, virus, spyware y demás problemas

► Con el número 1 se incluye un DVD con Debian Linux Sarge, un sistema Linux completo y en español que incluye más de 7000 aplicaciones.



**NVU**  
Diseño Web

**OpenOffice.org**  
Suite de Ofimática

**Thunderbird**  
Gestión de Correo

**Evolution**  
Suite de Comunicaciones

**Inkscape**  
Autoedición de Textos

**Firefox**  
Navegador Web

**Mono**  
Desarrollo .NET

**Gimp**  
Tratamiento y Retoque Digital

## Composición de la obra

12 coleccionables divididos en:

**Teoría**  
Conoce el sistema Linux a nivel profesional, cómo se instala en un PC, cómo se administra y cómo se trabaja con los principales entornos gráficos.

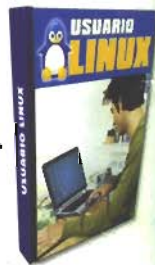
**Software**  
Descubre todas las aplicaciones libres que hay para Linux, desde la suite de ofimática OpenOffice.org a las herramientas de desarrollo compatibles con .NET.

**Tutorial Práctico**  
Aprende lo que realmente se necesita hoy en día: Cómo montar un servidor web, cómo configurar una red local, cómo programar una web dinámica o cómo instalar un cortafuegos profesional.

## Oferta de suscripción

**Usuario Linux** está disponible en tu quiosco, pero también puedes suscribirte directamente llamando al **91 628 02 03** y recibir la obra en tres entregas mensuales por solo **66,99** euros.

Además conseguirás de regalo la **tapa para encuadernar** la obra y una minisuscripción de tres números a la revista **Todo Linux**.



**Studio PRESS**

C/ del Río Ter, Nave 13 • Polígono Industrial "El Nogal" • 28110 Algete (Madrid) • Tel.: 916280203 • Fax: 916280935  
e-mail: [usuario@iberprensa.com](mailto:usuario@iberprensa.com) • [www.studiopress.es](http://www.studiopress.es)

**ya a la venta en quioscos y centros comerciales**